



DOCKET NO.: MSFT-1741/301923.01
Application No.: 10/600,178
Office Action Dated: January 29, 2008

PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of:

J. Kirk Haselden and Ashvini Sharma

Confirmation No.: 9748

Application No.: 10/600,178

Group Art Unit: 2193

Filing Date: June 2, 2003

Examiner: Tuan A. Vu

For: **Dependency Based Package Deployment**

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

DECLARATION PURSUANT TO 37 CFR § 1.131

We, the undersigned, hereby declare that:

1. We, J. Kirk Haselden and Ashvini Sharma, are the inventors of the invention described and claimed in U.S. Patent Application Number 10/600,178 (hereinafter referred to as "the 178 application"), filed June 20, 2003, in the United States Patent and Trademark Office.
2. We are aware that the pending claims of the 178 application have been rejected as being unpatentable over a publication entitled "Report Designer Component 9 – Creating a RDC Deployment Package," pp. 1-17 (hereinafter referred to as "RDC9").
3. We are aware that RDC9 was published on May 14, 2003.

Page 1 of 2

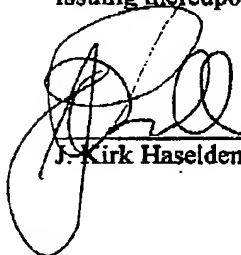
DOCKET NO.: MSFT-1741/301923.01
Application No.: 10/600,178
Office Action Dated: January 29, 2008

PATENT

4. In accordance with CFR § 1.131, as inventors of the subject matter of the pending claims, and without conceding the propriety of the rejections of the pending claims, we hereby declare that we invented the subject matter of the pending claims prior to May 14, 2003. We further hereby declare that we worked diligently from a date prior to May 14, 2003 to the date of constructive reduction to practice, June 20, 2003, the filing date of the 178 application, in order to prepare the 178 application and patent the invention.

5. Prior to May 14, 2003, internal documents entitled "DTS Deployment" and "Implementation of deployment executable" were prepared describing the subject matter of the 178 application. In support of the instant declaration, redacted copies of "DTS Deployment" and "Implementation of deployment executable" are attached hereto.

6. We hereby declare that all statements made herein of our own knowledge are true and that all statements made on information or belief are believed to be true; and further, that these statements were made with the knowledge that willful false statements and the like are punishable by fine or by imprisonment, or both, under § 1001 of Title 18 of the United States Code, and that such willful statements may jeopardize the validity of the application, any patent issuing thereupon, or any patent to which this verified statement is directed.


J. Kirk Haselden
4/25/08
Date


Ashvini Sharma
4/25/08
Date

DTS Deployment

Title	DTS Deployment
Author	[REDACTED]
Filename	Redacted Versio Of DTSDeployment.doc
Version	DRAFT
Created	[REDACTED]
Last update	[REDACTED]
Pages	4
Update history:	
[REDACTED]	Created initial draft

DTS Deployment.....	1
Owners:.....	2
Change history:.....	2
Review history:.....	2
Introduction:	3
General goals:.....	3
Scenarios:	3
Deploying a package with a Deployment Bundle.....	3
Deploying a package through SQL Server.....	3
The Five Levels of Deployment	3
None	3
Minimal	3
Partial	3
Full.....	4
Data	4
The Details	4
Cookbook	4
Open Issues	4

DTS Deployment

Owners:

Program Manager	
Development	
QA	
UE	
Product Design (UI)	n/a
Usability	n/a
Primary Reviewers	
Secondary Reviewers	
Current Status	Draft

Change history:

Date	Changes	By
	Created	

Review history:

Date	Changes	By
------	---------	----

Introduction:

One of the complaints about DTS 2000 is that it is difficult to deploy packages. Specifically related to this document, it is difficult to move a package from one machine to another. We need to provide a better way to move packages to other machines. The aim of this document is to describe the deployment architecture for DTS Yukon and how to use it.

This is a proposed solution and is an early draft. Your input is encouraged.

General goals:

The general deployment goals are:

1. Enable simple deployment of packages or data from one machine to another.
2. Enable deployment either through SQL Server or through a bundled deployment file.
3. Make the deployment process transparent to the user.
4. Deployment should be extensible to allow third parties to participate.
5. Deployment should be configurable, i.e. allow for parameters.
6. Deployment bundles should be self installing and require very little if any interaction with the user.

Scenarios:

Deploying a package with a Deployment Bundle

I have a package that uses the pipeline and a custom task that I created. I need to move the package to a new machine that has never had my custom task installed on it. I go to the machine where the package was installed and open the package that I want to deploy. I select the deploy menu item and a dialog box pops up with some options for deploying the package. Since the target machine has never had my custom task installed on it, I tell the deployment wizard which dependent tasks to include in the package. After I set the options I click OK. One of the options was the name of the deployment bundle I wish to create. When the deployment is complete, I go to the folder I specified and I see that the deployment bundle is there. I copy the deployment bundle over to the target machine. On the target machine, I right click the deployment bundle and click "Install". Then I open up the DTS Designer, open the deployed package and run it without errors.

Deploying a package through SQL Server

The scenario is the same, except I want to deploy everything through SQL Server.

[How is this different? Do we want to create a bundle and store it in binary form in the server? Do we just deploy the package through a remote insert and bundle the rest into a deployment file?]

The Five Levels of Deployment

None

None is what we have today. Essentially, the user copies the package file and hopes for the best. However, we should be even better with error reporting that indicates what package dependencies are missing.

Minimal

Minimal involves copying the package and a configuration file.

Partial

Partial includes a minimal bundle with the binaries for tasks that are

DTS Deployment

Full

A full deployment bundle combines a partial bundle with the redistributable portions of the runtime. This would make it possible for a fully bundled package to be deployed to a machine with no DTS installation and run with no errors.*

Data

A data bundle is a combination of any of the above bundles with other items also bundled such as SQL objects or tables.

* - Assuming no dependencies outside of DTS.

The Details

Ideas on how we would accomplish this.

For full deployment, we would leverage the work the setup folks are doing. There would need to be a well known location for the DTS redistributables that deployment would bundle with packages. There needs to be some thought put into how the runtime redistributables are factored and how the bundling of binaries is done.

- Do we always bundle the whole runtime?
- Do we only bundle what the user requests?
- For partial deployments do we bundle internal tasks if they are in the package or are these user settings?
- Can full deployment bundles be deployed through SQL Server?

Tasks can describe their dependencies through a property called "Dependencies". When deploying, the package enumerates all the tasks and interrogates this property. If the task is "marked" by the user for deployment, the package uses this property to discover dependencies needed to be bundled for that task.

Bundles are built using MSI files.

Cookbook

Open Issues

Who is doing what?

Bundling - Aqueduct Runtime Team?

Wizard - Picasso Designer Team?

Data bundling - Komodo Team?

The details need to be worked out.

"Voodoo doll" Jim wants to include signed and clear text features as well as the notion of a verify only deployment.

Implementation of deployment executable

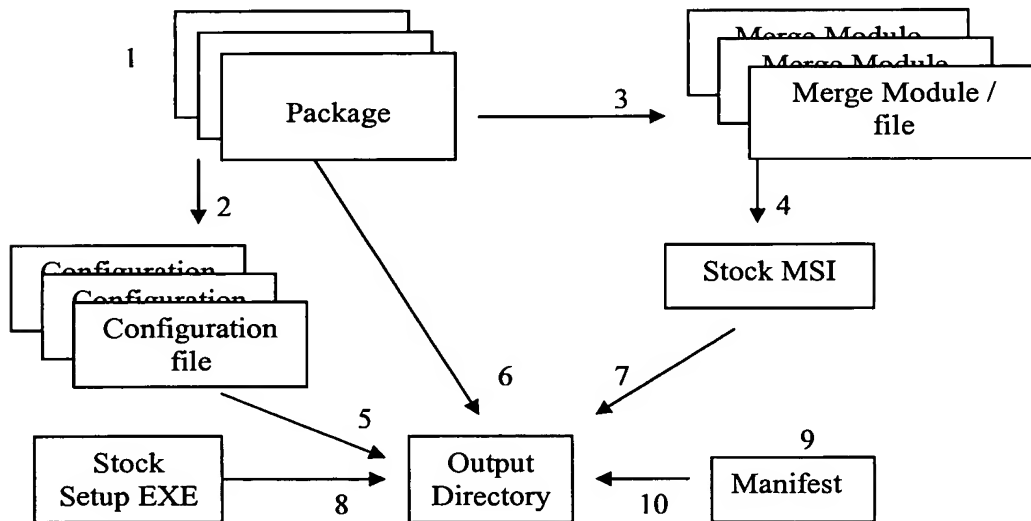
Setting deployment properties:

The following properties are exposed as project properties to facilitate setup of deployment options.

1. Create Deployment Utility -> Boolean. Default: False
2. Include dependencies in deployment utility -> Boolean. Default: True
3. Include DTS Runtime in deployment utility -> Boolean. Default: False
4. Allow configuration changes during deployment -> Boolean. Default: True
5. Deployment directory -> String. Default: \${OutputDir}/Deployment

Building the deployment exe

The following steps are taken during build time:



The steps are:

1. Packages information is received from the project.
2. Packages are inspected to create a list of Configuration files
3. If user has asked to also include dependencies:
 - a. VS' project interfaces are used for finding out dependencies of each package.
 - b. For each dependency, Deployment Project team's code is used to find any merge modules which must be included.

- c. If no merge modules are found, the dependencies are included as is.
- 4. Merge modules and dependency files are added to the stock MSI shell.
- 5. Packages are written out to the output directory
- 6. Configuration files are written out to the output directory
- 7. Stock MSI created from step #4 is written out to the output directory.
- 8. A stock setup EXE shell is written out to the output directory.
- 9. A manifest is created of all the files copied in steps 5 – 7 (inclusive). The manifest also contains other information about:
 - a. Creation time
 - b. Creation user
 - c. Project name
- 10. Manifest is copied to the output directory.

